

98 P245 L



19 BUNDESREPUBLIK
DEUTSCHLAND



DEUTSCHES
PATENTAMT

12 Übersetzung der
europäischen Patentschrift
87 EP 0481231 B1
10 DE 691 28 271 T2

51 Int. Cl.⁶:
G 06 F 11/00

B2

- 21 Deutsches Aktenzeichen: 691 28 271.4
- 86 Europäisches Aktenzeichen: 91 115 808.7
- 86 Europäischer Anmeldetag: 18. 9. 91
- 87 Erstveröffentlichung durch das EPA: 22. 4. 92
- 87 Veröffentlichungstag
der Patenterteilung beim EPA: 26. 11. 97
- 47 Veröffentlichungstag im Patentblatt: 28. 5. 98

DE 691 28 271 T 2

<p>30 Unionspriorität: 599178 17. 10. 90 US</p> <p>73 Patentinhaber: International Business Machines Corp., Armonk, N.Y., US</p> <p>74 Vertreter: Teufel, F., Dipl.-Phys., Pat.-Anw., 70569 Stuttgart</p> <p>84 Benannte Vertragsstaaten: DE, FR, GB</p>	<p>72 Erfinder: Smith, Donald M., Germantown, Maryland 20874, US</p>
--	--

54 Verfahren und System zur Erhöhung der Betriebsverfügbarkeit eines Systems von Rechnerprogrammen, wirkend in einem verteilten Rechnersystem

Anmerkung: Innerhalb von neun Monaten nach der Bekanntmachung des Hinweises auf die Erteilung des europäischen Patents kann jedermann beim Europäischen Patentamt gegen das erteilte europäische Patent Einspruch einlegen. Der Einspruch ist schriftlich einzureichen und zu begründen. Er gilt erst als eingelegt, wenn die Einspruchsgebühr entrichtet worden ist (Art. 99 (1) Europäisches Patentübereinkommen).

Die Übersetzung ist gemäß Artikel II § 3 Abs. 1 IntPatÜG 1991 vom Patentinhaber eingereicht worden. Sie wurde vom Deutschen Patentamt inhaltlich nicht geprüft.

DE 691 28 271 T 2

BESCHREIBUNG

Verfahren und System zur Erhöhung der Verfügbarkeit eines
Systems von Computerprogrammen in einem verteilten Computer-
system

Die im folgenden ausführlich beschriebene Erfindung bezieht sich auf Datenverarbeitungssysteme und insbesondere auf Systeme und Verfahren zur Verbesserung der Fehlertoleranz in Datenverarbeitungssystemen.

Die Verfügbarkeit wird wie folgt beschrieben: "Wenn ein Systemstimulus vom System verarbeitet wird und das System innerhalb einer zugeordneten Reaktionszeit ein richtiges Ergebnis für diesen Stimulus erzeugt und dieses Ergebnis für alle Stimuli zutrifft, beträgt die Verfügbarkeit des Systems 1.0."

Zu einer hohen Betriebsverfügbarkeit tragen zahlreiche Faktoren bei: (1) Fehler im Hardware- und Softwaresystem müssen mit ausreichend hoher Deckungsrate erkannt werden, um die Anforderungen zu erfüllen; (2) die Eigenverfügbarkeit der Hardware (als einfache numerische Verfügbarkeit seines Redundanznetzes), einschließlich interner und externer Redundanz, muß höher als die erforderliche Betriebsverfügbarkeit sein; und (3) Softwarefehler dürfen nicht sichtbar sein oder die betriebliche Nutzung des Systems nicht nachträglich beeinflussen. Die Erfindung beschäftigt sich mit dem dritten Faktor und geht dabei von der wichtigen Annahme aus, daß Softwarefehler aufgrund von Designfehlern und Hardwarefehlern häufig auftreten und fatal sind.

In der herkömmlichen Technik wurde versucht, dieses Problem durch doppelte Kopien der gesamten Softwarekomponente auf zwei oder mehr Einzelprozessoren und die Bereitstellung eines Mittels zur Übermittlung des "Gesundheitszustands" des aktiven

Prozessors an den Ersatzprozessor zu lösen. Wenn der Ersatzprozessor durch die Überwachung des Zustands des aktiven Prozessors feststellt, daß der Ersatzprozessor den Betrieb übernehmen muß, initialisiert er die gesamte gespeicherte Softwarekomponente, und der aktive Prozessor beendet seine Operationen. Das Problem bei dieser Verfahrensweise ist, daß das gesamte System an der Wiederherstellungsmaßnahme beteiligt ist. Das führt dazu, daß die Wiederherstellungszeiten lang sind und Fehler beim Wiederherstellungsprozeß das System normalerweise außer Betrieb setzen. Hinzu kommt, daß wenn die redundanten Kopien des Softwaresystems normal in Betrieb sind (eines als Schatten des anderen), sich der Effekt von Allgemeinmodusfehlern erheblich auswirkt und das ganze System beeinträchtigt.

Ein Ansatz zur Lösung des oben genannten Problems wird in den Protokollen des 9. Symposiums über zuverlässige verteilte Systeme vom 9.-11. Oktober 1990 in Alabama unter dem Titel "The Design and Implementation of a Reliable Distributed Operating System - ROSE" von T.P. NG beschrieben. In diesem Artikel wird ein modulares verteiltes Betriebssystem beschrieben, das den Aufbau zuverlässiger Anwendungen und die Verarbeitung von Hardwarefehlern unterstützt. Ziel ist, die Verfügbarkeit von Daten trotz dieser Fehler zu erhöhen. Dabei werden sogenannte replizierte Adreßraumobjekte (RAS-Objekte) bereitgestellt, auf deren Inhalt mit hoher Wahrscheinlichkeit zugegriffen werden kann. Des weiteren ermöglicht eine sogenannte flexible Prozeßabstraktion (RP-Abstraktion) den Benutzerprozessen, Hardwarefehler mit minimaler Unterbrechung zu überstehen. Im folgenden werden vor allem zwei verschiedene Implementierungen der flexiblen Prozesse vorgestellt: ein Prozeß, der die Information über seinen Zustand in einem RAS-Objekt regelmäßig prüft, und ein zweiter Prozeß, der mit wiederholter Ausführung arbeitet, indem derselbe Code zur gleichen Zeit in verschiedenen Knoten ausgeführt wird. "Modularität" des verteilten Betriebssystems bedeutet in diesem Sinn, daß die Kernschicht des Betriebssystems zahlreiche

Prozesse mit unterschiedlichen Adreßräumen unterstützt. Innerhalb jedes Prozesses und seines Adreßraums können mehrere Aufgaben parallel ausführen und über einen geteilten Speicher miteinander kommunizieren. Mehrere Prozesse können replizieren und einen Teil ihres Adreßraums teilen. Mehrere Prozesse replizieren die Ausführung einer Anwendung, um ein Bild eines einzelnen, aber sehr zuverlässigen Prozesses bereitzustellen. Die Anwendungen können zwei Prozesse aufstellen, die zum selben Konfigurationsobjekt gehören und ein RAS-Objekt teilen. Bei einem Fehler in einem der Prozesse konfiguriert sich der zweite Prozesse neu, um den ersten Prozeß aus der Konfiguration zu entfernen. Wenn die Umkonfigurierung erfolgreich ist, liest der zweite Prozeß den Inhalt des RAS-Objekts und nimmt die Verarbeitung an dem Punkt wieder auf, an dem der erste Prozeß aufgehört hat.

Gegenstand der Erfindung ist es daher, die Verfügbarkeit eines Systems von Computerprogrammen in einem verteilten Computersystem zu erhöhen, bei dem die Wiederherstellung nach einem Software- oder Hardwarefehler erfolgt, bevor der Fehler im System sichtbar wird.

Ein weiterer Gegenstand der Erfindung ist es, eine Fehlertoleranz in einem System mit Computerprogrammen in einem verteilten Computersystem mit hoher Verfügbarkeit und schneller Wiederherstellungszeit bereitzustellen.

Ein weiterer Gegenstand der Erfindung ist es, eine bessere Verfügbarkeit in einem System mit Computerprogrammen in einem verteilten Computersystem zu gewährleisten, wobei die Software weniger komplex als bei der herkömmlichen Technik ist.

Diese Gegenstände, Merkmale und Vorteile der Erfindung sind durch das in Anspruch 1 beschriebene Verfahren gekennzeichnet. Die Erfindung beschreibt einen Mechanismus zur Organisation der

Computersoftware, die nach einem Fehler (der Software selbst oder der Hardware) wiederhergestellt wird, noch bevor der Fehler sichtbar wird. Die Software sorgt mit anderen Worten dafür, daß sie nach einem Fehler wiederhergestellt wird und den Stimulus verarbeitet oder ablehnt, so daß der Benutzer des Systems das Ergebnis innerhalb der für diese Art Stimulus vorgesehenen Reaktionszeit erhält.

Eine Softwarestruktur, die als Betriebseinheit (OU) bezeichnet wird, und eine dazugehörige Verfügbarkeitsmanagementfunktion (AMF) sind die wichtigsten Bestandteile der Erfindung. Im folgenden werden die OU und Teile der AMF beschrieben. Das OU-Konzept sieht so aus, daß die Systemsoftware so weit wie möglich in unabhängige Module unterteilt wird, die über einen Netzwerk-Server miteinander in Wechselwirkung treten. Ein Stimulus erreicht das System und wird zum ersten Modul in der Verkettungsstruktur geleitet, von wo er alle erforderlichen Module durchläuft, bis die entsprechende Reaktion erfolgt und dem Benutzer des Systems bereitgestellt wird.

Jedes Modul besteht eigentlich aus zwei Kopien des Codes und Datenraums der OU. Eine der Kopien, der primäre Adreßraum (PAS), enthält die aktuellen Zustandsdaten. Die andere Kopie, der Ersatzadreßraum (SAS), läuft auf einem anderen Prozessor und kann aktuelle Zustandsdaten enthalten oder nicht, wie noch an späterer Stelle zu beschreiben sein wird.

Die Verfügbarkeitsmanagementfunktion (AMF) steuert die Zuweisung von PAS- und SAS-Komponenten zu den Prozessoren. Wenn die AMF einen Fehler feststellt, wird ein SAS zu einem PAS und der ursprüngliche PAS wird abgeschlossen. Die Daten-Server im Netzwerk werden ebenfalls über die Änderung informiert, so daß die gesamte Kommunikation zum neuen PAS umgeleitet wird. Auf diese Art und Weise kann die Systemverfügbarkeit aufrechterhalten werden.

Diese und andere Gegenstände, Merkmale und Vorteile der Erfindung werden anhand der Begleitzeichnungen noch deutlicher.

Fig. 1 ist ein Zeitdiagramm mit Reaktionszeitzuweisungen.

Fig. 2A ist ein schematisches Diagramm mehrerer Betriebseinheiten in einem Netzwerk.

Fig. 2B ist ein anderes Diagramm der Architektur der Betriebseinheit, das die Verwendung einer Daten-server-OU durch mehrere Anwendungen zeigt.

Fig. 2C zeigt die Architektur der Betriebseinheit.

Fig. 2D zeigt die Architektur der Betriebseinheit mit einem Beispiel der Zuweisung allgemeiner Betriebseinheiten in drei Gruppen.

Fig. 3A-F zeigen die Betriebseinheit in verschiedenen Stufen während der Initialisierung und des Betriebs, der Umkonfigurierung und Wiederherstellung.

Fig. 1 zeigt ein Zeitdiagramm mit Reaktionszeitzuweisung. Die benötigte Reaktionszeit (T_{\max}) zwischen einer Stimulus-Eingabe und der erforderlichen Ausgabe wird zugewiesen, so daß ein Teil davon für die normale Erzeugung der Reaktion (T_{normal}) zur Verfügung steht, ein Teil für eine unvorhergesehene Betriebsmittelbelegung (T_{Belegung}) und ein Teil für die Wiederherstellung nach Fehlern ($T_{\text{Wiederherstellung}}$). Der erste Teil, T_{normal} , wird unter den Software- und Hardwareelementen des Systems nach deren Verarbeitungsanforderungen aufgeteilt. Diese Zuweisung bestimmt die von jeder Hardware- und Softwarekomponenten des Systems benötigte Leistung. Der zweite Teil, T_{Belegung} , wird nicht zugewiesen. Der letzte Teil, $T_{\text{Wiederherstellung}}$, ist ausreichend lang, so daß genügend Zeit für die

Fehlererkennung (einschließlich Auslassungsfehlern), Hardware-Umkonfigurierung, Software-Umkonfigurierung und die erneute Erzeugung einer erforderlichen Reaktion vorhanden ist. Als Faustregel gilt, daß die benötigte Reaktionszeit T_{max} geteilt wird und anschließend die erste Hälfte noch einmal geteilt wird, so daß ein Viertel der benötigten Reaktionszeit für die Erzeugung der normalen Reaktion T_{normal} zur Verfügung steht, während das zweite Viertel für eine unvorhergesehene Betriebsmittelbelegung, $T_{Belegung}$, bereit steht. Die zweite Hälfte der Reaktionszeit, $T_{Wiederherstellung}$, steht dann für die Fehlererkennung und Reaktionserzeugung zur Verfügung.

Die Erfindung beschäftigt sich vor allem mit der speziellen Frage, wie die für die Hardware- und Software-Umkonfigurierung benötigte Zeit, $T_{Wiederherstellung}$, eines komplexen Systems auf einen Bruchteil von T_{max} reduziert werden kann. Dieses Problem wird durch eine Softwarestruktur gelöst, die als Betriebseinheit (OU) bezeichnet wird, und eine dazugehörige Verfügbarkeitsmanagementfunktion (AMF). Im folgenden werden nun die OU und Teile der AMF beschrieben.

In Fig. 2A sieht das OU-Konzept so aus, daß die Systemsoftware so weit wie möglich in unabhängige Module oder OU 10 unterteilt wird, die über einen Netzwerk-Server 12 miteinander in Wechselwirkung treten. Keines der Module teilt Dateien mit einem anderen Modul, und keines der Module geht davon aus, daß ein anderes Modul sich in derselben Maschine wie es selbst befindet (oder nicht). Ein Stimulus 14 erreicht das System und wird zum ersten Modul in der Verkettungsstruktur weitergeleitet, von wo aus es alle erforderlichen Module durchläuft, bis eine entsprechende Reaktion erzeugt und dem Benutzer 16 des Systems zur Verfügung gestellt wird.

Jedes Modul speichert sämtliche notwendigen Zustandsdaten für seinen eigenen Betrieb. Wenn zwei oder mehr Module Zugriff auf

dieselben Zustandsdaten wollen, (1) muß jedes diese Daten enthalten, (2) müssen Aktualisierungen dieser Daten als normale Verarbeitungsoperationen unter den Modulen übermittelt werden, oder (3) muß jedes tolerant gegenüber möglichen Unterschieden zwischen seinen Zustandsdaten und den Daten der anderen Module sein. Diese Toleranz kann je nach Anwendung verschieden aussehen und kann Mechanismen zur Erfassung und zum Ausgleichen (oder zur Korrektur) von Zustandsunterschieden beinhalten. Wenn zwei Module 10 unbedingt eine gemeinsame Datenbasis für die Verarbeitung oder andere Gründe haben müssen, sind sie nicht "unabhängig" und werden im Rahmen dieser Erfindung zu einem Modul zusammengefaßt.

Es ist möglich, daß ein Modul 10' Datenserver-Funktionen für zahlreiche andere Module durchführt (siehe Fig. 2B), unter der Voraussetzung, daß diese anderen Module 10 Fehler und Verlust der Server-Funktion ausgleichen können. Ausgleichen heißt in diesem Fall, daß sie ihren Clients weiterhin ihre wichtigen Dienste anbieten, und daß der nicht mögliche Zugriff auf den gemeinsamen Zustand nicht zu einer unannehmbaren Warteschlange oder der Unterbrechung des Betriebs führt. Dadurch sind die Verwendungsmöglichkeiten dieser gemeinsamen Server natürlich eingeschränkt.

Ein Modul 10 muß schließlich vordefinierte verschlechterte oder andere Betriebsmodi für den Fall bereitstellen, daß die Dienste eines anderen Moduls benötigt werden, dieses Modul jedoch nie zur Verfügung steht. Eine Ausnahme bildet der Fall, wenn ein Server Teil eines Prozessors ist (wenn er mit der Zuweisung oder Verwendung von Prozessorbetriebsmitteln zu tun hat), dann kann das Modul nämlich unbedingte Abhängigkeiten vom Server haben. In diesem Fall behandelt die Verfügbarkeitsmanagementfunktion einen Fehler des Server als Fehler des Prozessors. Wenn ein Modul die oben genannten Bedingungen erfüllt, wird es durch die folgende neue Struktur seiner Daten, seiner Logik und

seiner Rolle im System zur OU. Diese Struktur ist in Fig. 2C dargestellt.

Zwei vollständige Kopien der Module werden in zwei unabhängige Adreßräume 20, 20' in zwei separate Computer 22, 22' geladen. Eines dieser Module kennt der Netzwerk-Server als den primären Adreßraum (PAS), zu dem sämtliche Dienstanforderungen der anderen Module gesendet werden. Das andere Modul ist der Ersatzadreßraum (SAS), den nur der PAS kennt. Alle anderen Module kennen diesen Adreßraum nicht. Der PAS sendet anwendungsabhängige Zustandsdaten zum SAS, so daß der SAS den Zustand des PAS kennt. Ob die Schnittstelle zwischen dem PAS und dem SAS einer OU eine synchrone Festschreibung oder eine nicht synchronisierte Schnittstelle ist, legt die Erfindung nicht fest; es findet ein Kompromiß zwischen der stabilen Reaktionszeit und der für einen neu umgestuften PAS benötigten Zeit, um sich mit seinen Servern und Clients zu synchronisieren, statt. Dieser Kompromiß wird nachfolgend in Strategie 1 beschrieben.

Der PAS hält den Zustand, der für die normale Anwendungsverarbeitung durch das Modul benötigt wird. Der SAS verfügt über ausreichende Zustandsdaten, so daß er im Bedarfsfall zum PAS werden kann. Der benötigte Datenumfang ist dabei von der Anwendung abhängig und nicht Gegenstand dieser Erfindung.

Sowohl der PAS als auch der SAS einer OU haben offene Sitzungen mit allen Servern der OU. Die SAS-Sitzungen werden nicht benötigt, bis der SAS zum PAS wird. Wenn die AMF den SAS veranlaßt, die Rolle des PAS zu übernehmen, stellt er sicher, daß sein gegenwärtiger Zustand selbstkonsistent ist (ein Fehler kann sich dadurch ergeben haben, daß der PAS nur einen Teil der betreffenden Aktualisierungsmeldungen gesendet hat), und kommuniziert anschließend mit den Clients und Servern des PAS, um eine Zustandsübereinstimmung herzustellen. Die Folge kann sein, daß die Zustandsdaten des SAS aktualisiert werden oder die Zu-

standsdaten der Clients und Server wiederholt werden. Jede Wiederholung muß mit der erneuten Verarbeitung eines beeinträchtigten Stimulus und/oder der Informierung der Benutzer, daß die Stimuli ignoriert worden sind, einhergehen. Gleichzeitig wird der Netzwerk-Server aktualisiert, so daß er alle neuen oder in Warteschlangen befindlichen Dienstanforderungen zum SAS anstatt zum PAS sendet. Mit der letztgenannten Maßnahme wird der SAS zum PAS gemacht, und anschließend wird ein neuer SAS in einem anderen Prozessor als dem, in dem sich der neue PAS befindet, gestartet.

Laut Erfindung sind mehrere Strategien zur Aufrechterhaltung der Ersatzdaten durch den PAS von Bedeutung. Sie werden im folgenden kurz zusammengefaßt.

Strategie 1. Der SAS kann eine komplette Kopie des PAS-Zustands enthalten. Wenn die Kopie im SAS festgeschrieben wird, bevor der PAS auf seinen Stimulus reagiert, ist der Wiederanlauf sehr schnell, doch die Reaktionszeit für alle Stimuli dauert am längsten. Dieser Ansatz wird besser, wenn es die Reaktionszeitanforderungen erlauben.

Strategie 2. Der SAS kann eine "Nachfolge"-Kopie des PAS-Zustands enthalten. Der PAS sendet in diesem Fall Zustandsaktualisierungen so wie sie auftreten, am Ende der Verarbeitung eines Stimulus, oder stapelt sie für mehrere Stimuli. Der SAS nimmt den jeweiligen PAS-Zustand auf und muß daher die Zustandskonsistenz in seinen Daten und zwischen ihm und seinen Servern und Clients prüfen. Dies führt zu einer sehr schnellen stabilen Reaktionszeit, doch ist die Verarbeitung während der Fehlerbehebung langsam.

Strategie 3. Der SAS kennt die gerade vom PAS verarbeiteten Stimuli, so daß ihm der genaue Zustand der Beziehungen zwischen dem PAS und seinen Clients und Servern bekannt ist. Zu diesem

Zweck ist die Festschreibung von Anfang und Ende einer Transaktion zwischen PAS und SAS erforderlich, doch die während der Fehlerbehebung erforderliche Synchronisierung innerhalb der OU wird verringert.

Diese Mechanismen können allein oder in verschiedenen Kombinationen innerhalb einer OU Anwendung finden. Die Wahl hängt von der Art der Stimuli ab, den Reaktionszeitanforderungen und dem von der Anwendung gehaltenen Zustand.

Die Verfügbarkeitsmanagementfunktion (AMF)

Bei den oben beschriebenen Eigenschaften ist zwar eine OU vorhanden, doch eine Verfügbarkeit erreichen diese allein nicht. Die Verfügbarkeit ergibt sich, indem diese OU-Architektur mit einer AMF kombiniert wird, die den Zustand aller OUs im System steuert. Die AMF verfügt über drei Komponenten, die jeweils eigene Rollen bei der Aufrechterhaltung einer hohen Verfügbarkeit der System-OUs spielen. Die Beziehung zwischen einer OU und der AMF ist in den Figuren 3A bis 3F dargestellt und wird im folgenden beschrieben.

Die wichtigste AMF-Funktion ist der Gruppenmanager. Eine Gruppe ist eine Reihe von ähnlich konfigurierten Prozessoren, die sich im Netzwerk befinden, um eine zuvor festgelegte Serie von einer oder mehreren OUs aufzunehmen. Jede Gruppe im System (sofern mehr als eine Gruppe vorhanden ist) wird unabhängig von den anderen Gruppen verwaltet. In Fig. 2D sind drei Gruppen dargestellt. In diesem Fall sind die OUs in jeder Gruppe (und nicht die Prozessoren) abgebildet.

Die Zahl der Prozessoren in jeder Gruppe und die Zuordnung von OU-PAS- und SAS-Komponenten zu diesen Prozessoren können je nach Verfügbarkeitsanforderung, Leistung jedes Prozessors und den Verarbeitungsbedürfnissen jeder PAS- und SAS-Komponente er-

heblich variieren. Die einzige Bedingung der Erfindung ist, daß sich der PAS und SAS einer OU in zwei getrennten Prozessoren befinden müssen, wenn ein Schutz vor Prozessorfehlern bestehen soll.

Gruppenmanagement: In Fig. 3A befindet sich der Gruppenmanager in einem der Prozessoren einer Gruppe (der Prozessor wird von einem nicht in dieser Erfindung vorgestellten Protokoll bestimmt). Er initialisiert die Gruppe für Kaltstarts und rekonfiguriert die Gruppe im Bedarfsfall für eine Fehlerbehebung. Innerhalb der Gruppe befinden sich der PAS und SAS jeder OU in separaten Prozessoren, wobei genügend Prozessoren vorhanden sind, um die internen Verfügbarkeitsanforderungen zu erfüllen. Der Gruppenmanager überwacht die Leistung der Gruppe als Einheit und koordiniert die Erfassung und Behebung aller Fehler innerhalb der Gruppe, die nicht auf Systemebene behandelt werden müssen. Diese Koordinierung beinhaltet:

1. Steuerung der Übernahme der funktionalen Zuständigkeit einer OU durch den SAS, wenn festgestellt wird, daß ein Fehler entweder im PAS oder im Prozessor oder im entsprechenden, für die Operation des PAS erforderlichen Betriebsmittel aufgetreten ist.
2. Initiierung des Anlegens eines neuen Adreßraums für einen neuen SAS, nachdem ein früherer SAS zum PAS geworden ist.
3. Aktualisierung des Bildes des Netzwerk-Servers vom Standort jeder OU, wenn ihr PAS unter Prozessoren einer Gruppe als Reaktion auf Fehler oder das geplante Ausschalten einzelner Gruppenprozessoren wandert.

Der Gruppenmanager stellt Fehler wie Prozessorfehler (durch Überwachungsprotokolle zwischen Gruppenmitgliedern) und Fehler einer OU fest, die sowohl den PAS als auch den SAS beein-

trächtigen, zum Beispiel Fehler, die durch ein fehlerhaftes Design bei der Kommunikation zwischen den beiden oder der gemeinsamen Verarbeitung von Ersatz-/Sicherungszustandsdaten auftreten. Die Mechanismen der Fehlererfassung werden in dieser Erfindung nicht behandelt.

Lokales Management: Die Unterstützung der OU-Architektur auf Gruppenebene hängt vom Vorhandensein bestimmter Funktionen in jedem Prozessor der Gruppe ab. Diese Funktionen werden als lokaler Manager 32 der AMF bezeichnet. Der lokale Manager ist in jedem Prozessor als Erweiterung des Steuerprogramms des Prozessors implementiert und ist für die Erfassung und Korrektur von Fehlern zuständig, die ohne Beteiligung einer höheren Ebene (Gruppe oder darüber) behoben werden können. Der lokale Manager hält Überwachungsprotokollkommunikationen mit jedem PAS oder SAS der OU in seinem Prozessor aufrecht, um nach Anormalitäten in ihrer Leistung Ausschau zu halten. Ferner erhält er eine Meldung vom Betriebssystem über ein auf der Maschinenebene in der Hardware oder Software festgestelltes Problem. Jedes Problem, das nicht lokal behoben werden kann, wird an den Gruppenmanager zur Lösung weitergegeben.

Globales Management: Die Isolierung und Korrektur von Problemen auf Systemebene und Problemen im Zusammenhang mit dem Netzwerk sind Aufgabe des globalen Managers 34 der AMF. Der globale Manager 34 korreliert Fehler und Zustände auf niedrigeren Ebenen, um Fehler im Netzwerkverhalten, Gruppenverhalten und Reaktionszeiten von Verkettungsstrukturen bestehend aus zahlreichen Prozessoren zu erfassen und zu isolieren. Ferner tritt er mit dem Benutzer des Systems in Verbindung, um Fehler zu beheben, die die Automation nicht beseitigen kann. Der globale Manager kann an jeder Station im Netzwerk arbeiten und ist selbst eine OU. Das Wandern der globalen Manager-OU 34 von einer Gruppe zur anderen wird bei Bedarf vom Benutzer durch Funktionen initiiert und überwacht, die im lokalen Manager in

jeder Prozessorstation enthalten sind.

Netzwerkmanagement: Der Netzwerkmanager ist Teil des globalen Managers 34 der AMF. Seine Funktionalität und sein Design hängen von der Konfiguration ab und sind nicht Gegenstand der vorliegenden Erfindung.

Die Figuren 3A-F zeigen die Funktionalität der AMF während der Initialisierung, des Betriebs, der Rekonfiguration und der Wiederherstellung des Systems. In Fig. 3A werden der PAS und SAS geladen und initialisiert. Die OU-Standorte werden in den Netzwerk-Server eingegeben. Fig. 3B zeigt die Synchronisierung des PAS und SAS mit den Clients und Servern, um eine Zustandskonsistenz zu erzielen. Fig. 3C zeigt eine stabile Zustandsoperation, bei der der PAS auf einen Stimulus reagiert und eine Reaktion erzeugt. Der SAS wird vom PAS aktualisiert.

Fig. 3D zeigt, was passiert, wenn der PAS ausfällt. Der alte SAS wird zum neuen PAS, und ein neuer SAS wird geladen und initialisiert. Die Netzwerk-Server werden ebenfalls mit dem neuen Standort der OU aktualisiert. Fig. 3E zeigt die Resynchronisierung des neuen PAS mit den Clients und Servern. Im stabilen Zustand (Fig. 3E) reagiert der neue PAS normal auf Stimuli.

Die Erfindung beschreibt eine Kombination der OU-Struktur und den Wiederherstellungsfunktionen der AMF. Herkömmliche Strategien und Mechanismen zur Software-Fehlerbehebung/Umschaltung beziehen sich zum Vergleich auf Einheiten von ganzen Prozessoren und nicht auf kleinere Software-Einheiten wie in dieser Erfindung. Da die Fehler- und Wiederherstellungseinheit, auf die die Erfindung abzielt, vergleichsweise klein ist, ist auch die Wiederherstellungszeit kurz. Hinzu kommt, daß die Wiederherstellung nach Fehlern auf Prozessezebene in kleinen Schritten in mehreren Prozessoren er-

18.12.97

- 14 -

folgen kann (in allen Prozessoren, die zur Aufnahme von SAS und PAS benutzt werden, die im Prozessor mit dem Fehler enthalten sind). Dies ist für die Aufrechterhaltung einer hohen Verfügbarkeit in großen Echtzeitsystemen von ausschlaggebender Bedeutung.

BT 990 004

ANSPRÜCHE

1. Ein Verfahren zur Erhöhung der Betriebsverfügbarkeit eines Systems von Computerprogrammen in einem verteilten Computersystem, das folgende Schritte umfaßt:

die Unterteilung eines Computerprogramms in zahlreiche unabhängige funktionale Module (10), die über einen oder mehrere Server (12), die im verteilten System miteinander verbunden sind, untereinander in Wechselwirkung treten, wobei keines der funktionalen Module Daten mit einem anderen Modul teilt, und wobei alle funktionalen Module sämtliche notwendigen Zustandsdaten für ihren eigenen Betrieb speichern;

die Steuerung des Zustands aller funktionalen Module (10) durch Verfügbarkeitsmanagementmittel, die die Leistung der funktionalen Module überwachen und die Erfassung und Behebung von Systemfehlern koordinieren;

das Laden einer ersten Kopie eines funktionalen Moduls in einen Adreßraum (20) eines ersten Prozessors (22) und das Versetzen einer zweiten Kopie des funktionalen Moduls in einen Adreßraum (20') eines zweiten Prozessors (22'), wobei beide Adreßräume physisch getrennt und unabhängig voneinander sind, und der Adreßraum des zweiten Prozessors nur dem entsprechenden ersten Prozessor bekannt ist;

wobei der erste Prozessor das erste funktionale Modul ausführt, um anwendungsabhängige Zustandsdaten zum zweiten Prozessor zu senden, wo diese vom zweiten funktionalen Modul empfangen werden, das auf dem zweiten Prozessor ausgeführt wird;

wobei der erste Prozessor das erste Modul ausführt und

einen normalen Anwendungsverarbeitungszustand hält, und der zweite Prozessor das zweite Modul ausführt und ausreichende Daten über den zweiten Zustand hält, die es ihm ermöglichen, zum ersten funktionalen Modul zu werden;

wobei der erste Prozessor, der das erste Modul ausführt, offene Sitzungen mit zahlreichen der Server hat, die mit ihm verbunden sind, und der zweite Prozessor, der das zweite Modul ausführt, zahlreiche offene Sitzungen mit allen Servern im Netzwerk hat;

wobei das zweite funktionale Modul als Reaktion auf einen Stimulus, der es veranlaßt, die Rolle des ersten funktionalen Moduls zu übernehmen, prüft, daß sein aktueller Zustand konsistent mit dem aktuellen Zustand des ersten funktionalen Moduls ist, worauf das zweite Modul mit allen Servern im Netzwerk kommuniziert, um eine Synchronisierung mit dem Zustand der Server herzustellen, wonach das erste funktionale Modul aufgegeben und die Server mit dem neuen Adreßraum aktualisiert werden;

wobei alle im Netzwerk angeschlossenen Clients und Server auf das zweite Modul reagieren, das die Rolle des ersten Moduls übernommen hat, indem alle neuen oder in Warteschlangen befindlichen Dienstanforderungen zum zweiten Modul anstatt zum ersten Modul geleitet werden;

wobei das zweite Modul die Rolle des ersten Moduls übernimmt, indem es Operationen eines primären Adreßraums (20) durchführt.

2. Das Verfahren nach Anspruch 1, bei dem das erste Modul und das zweite Modul synchron kommunizieren.
3. Das Verfahren nach Anspruch 1, bei dem das erste Modul und

das zweite Modul asynchron kommunizieren.

4. Das Verfahren nach Anspruch 1, bei dem das zweite Modul eine vollständige Kopie des Zustands des ersten Moduls hält.
5. Das Verfahren nach Anspruch 1, bei dem das zweite Modul eine Nachfolgekopie des Zustands des ersten Moduls hält.
6. Das Verfahren nach Anspruch 1, bei dem das zweite Modul die Stimuli kennt, die gerade vom ersten Modul verarbeitet werden.

FIG. 1

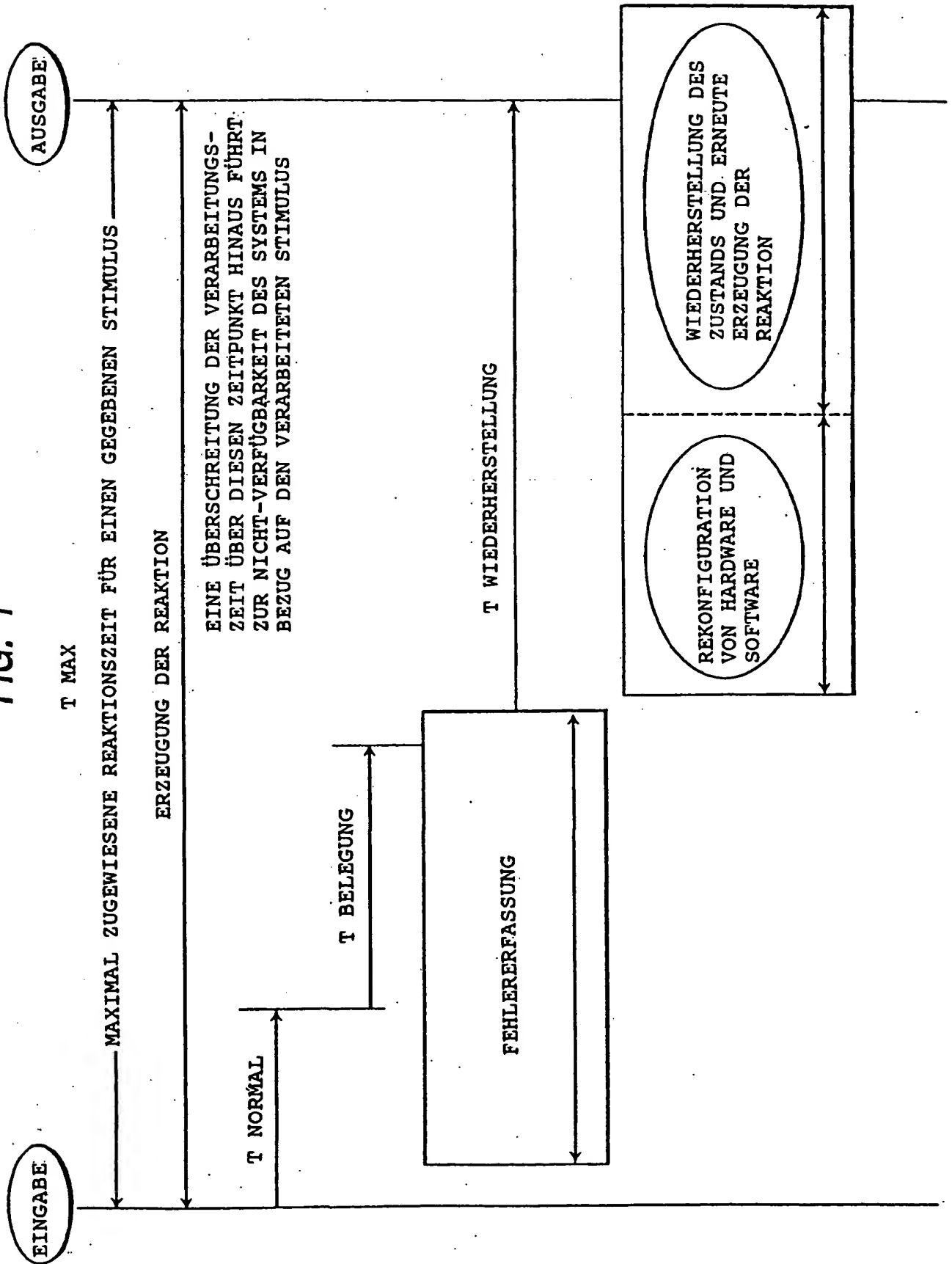


FIG. 2A

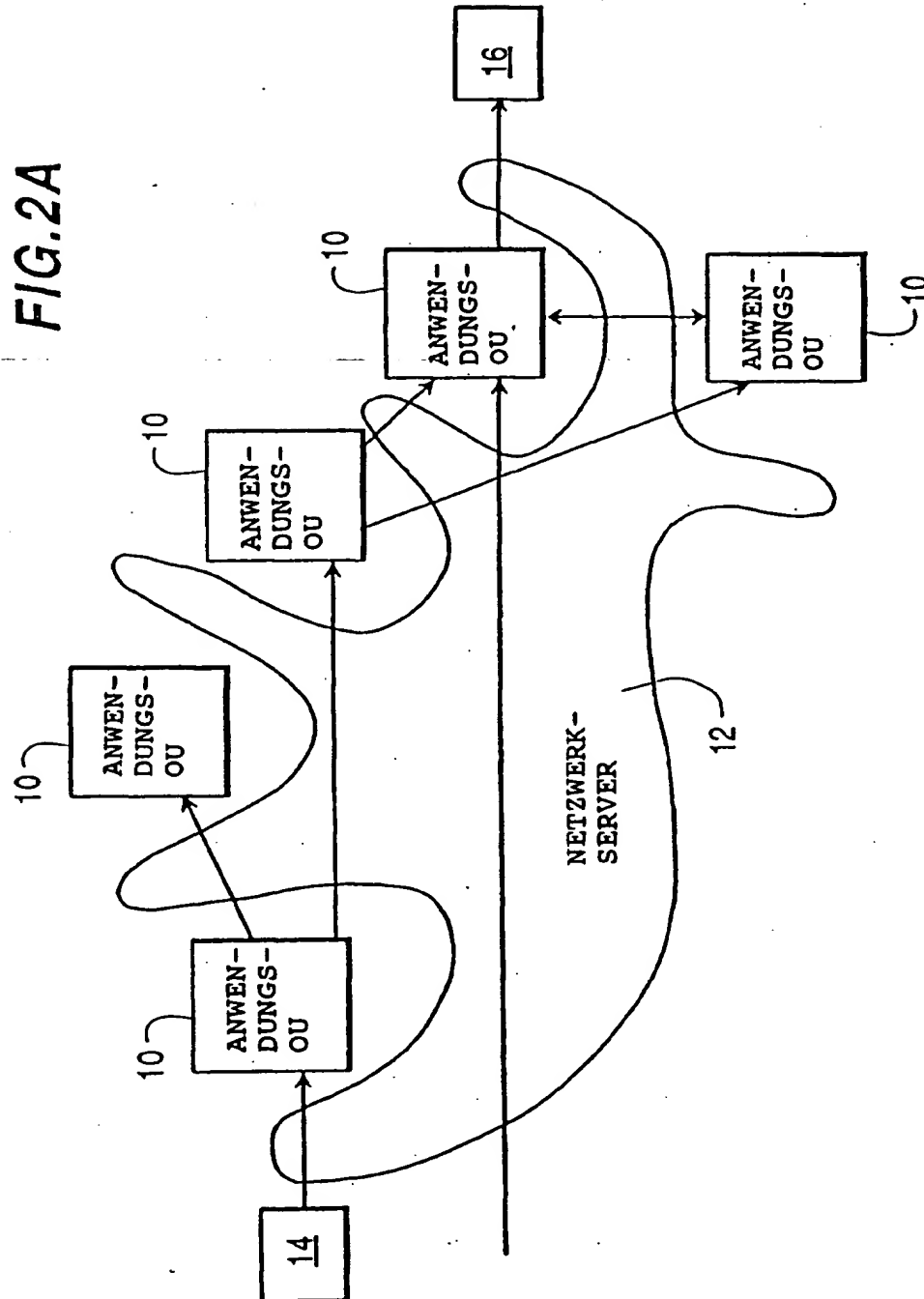


FIG.2B

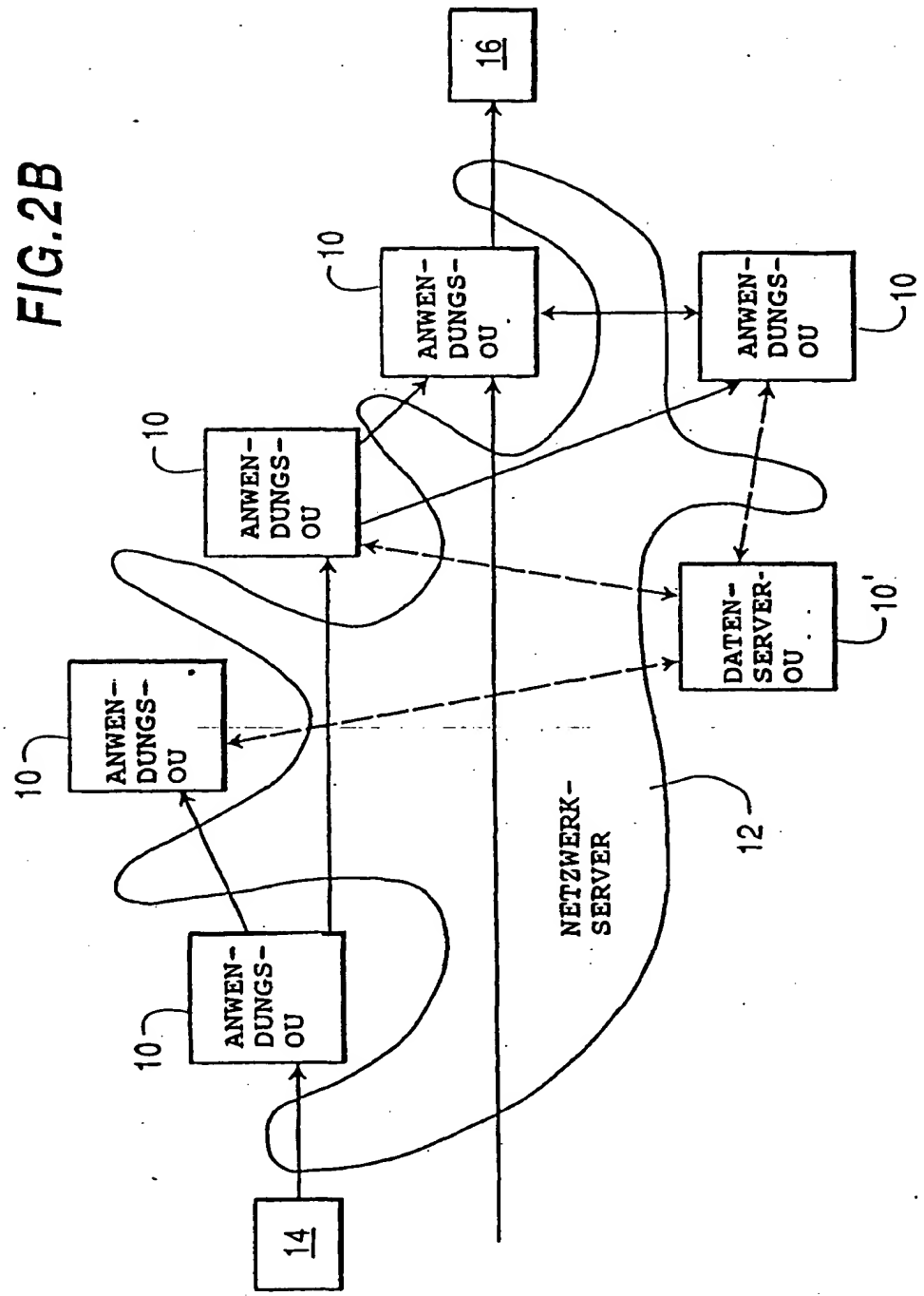
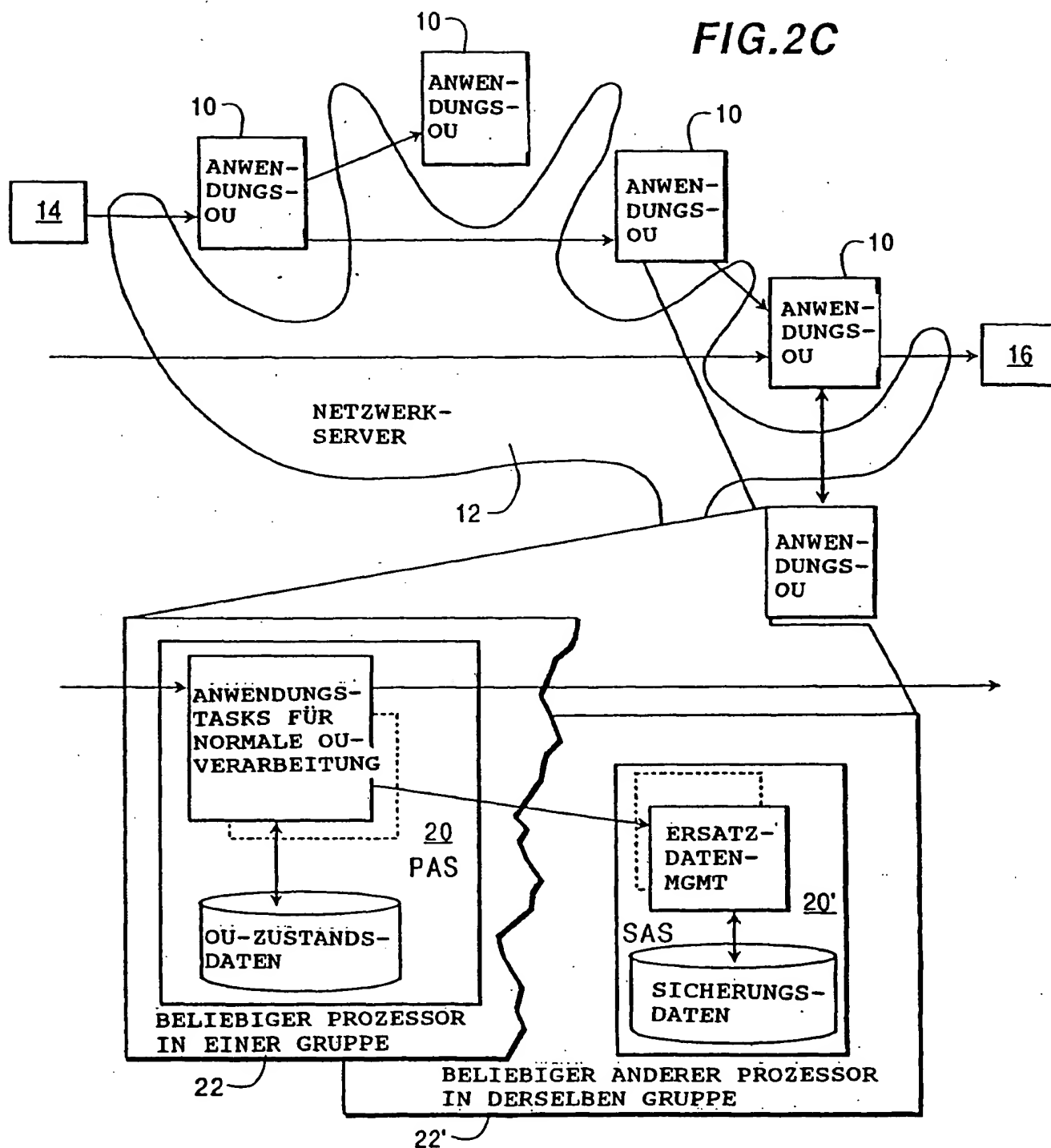


FIG.2C



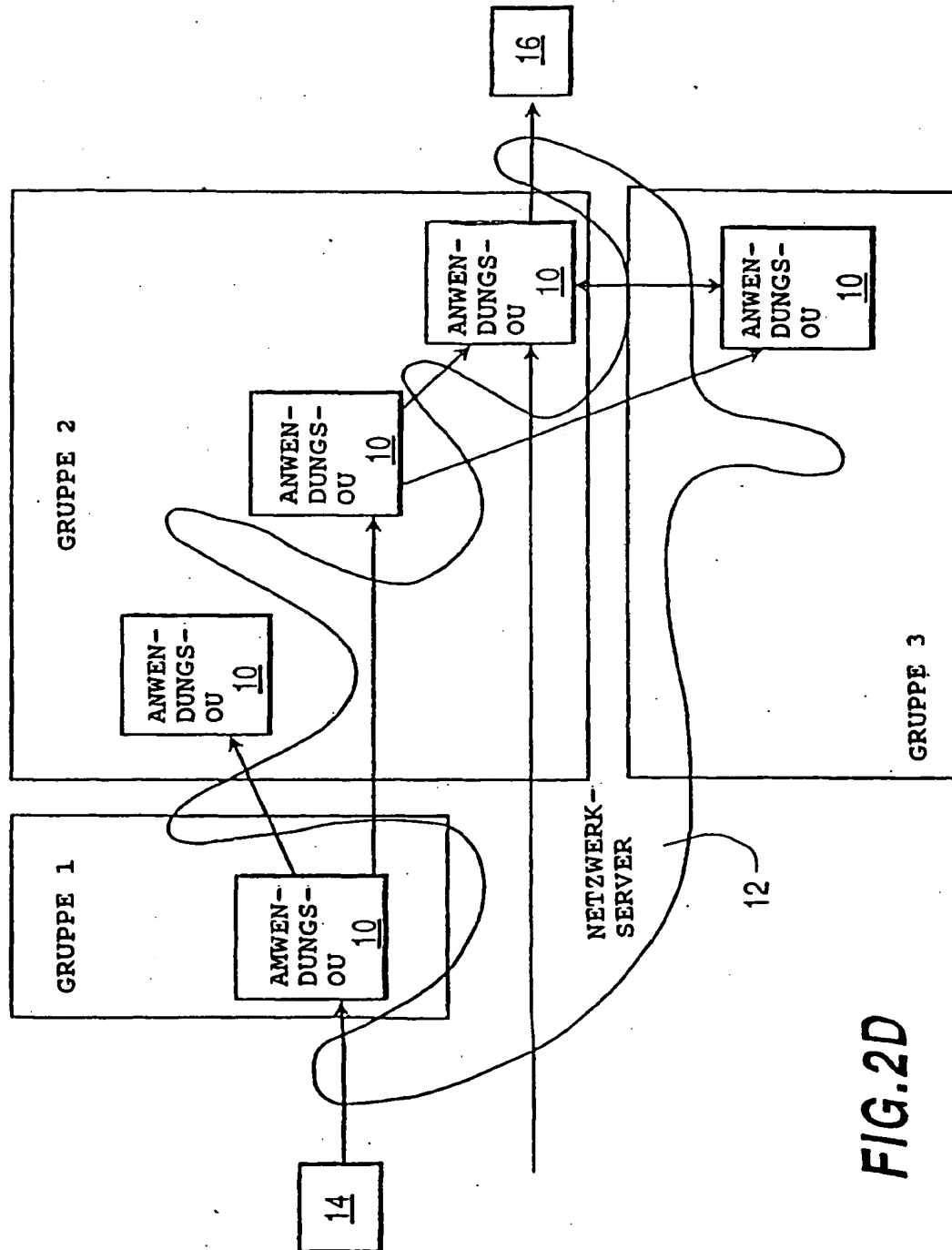


FIG.2D

FIG. 3A

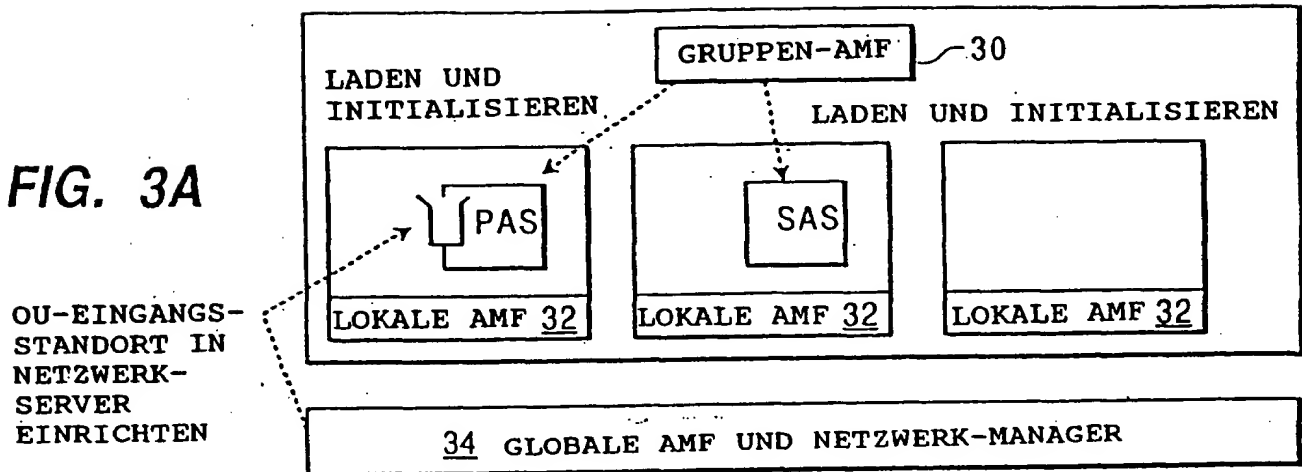


FIG. 3B

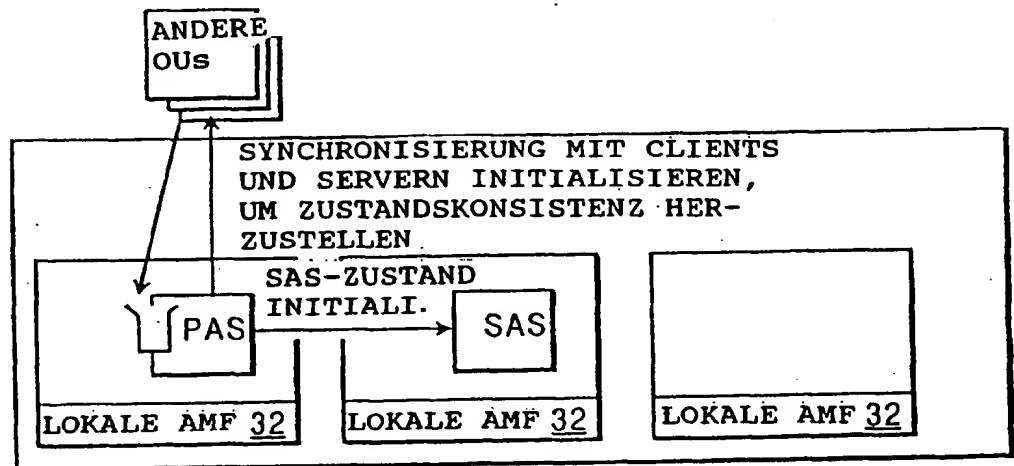


FIG. 3C

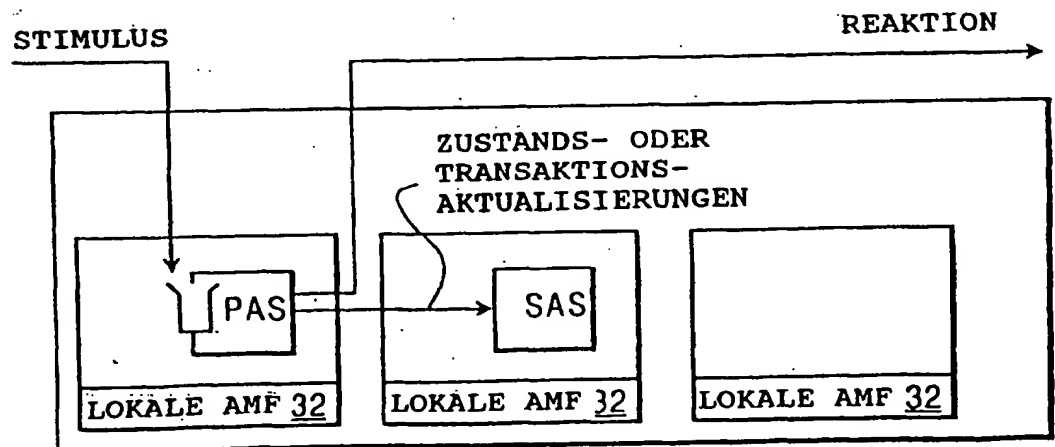


FIG. 3D

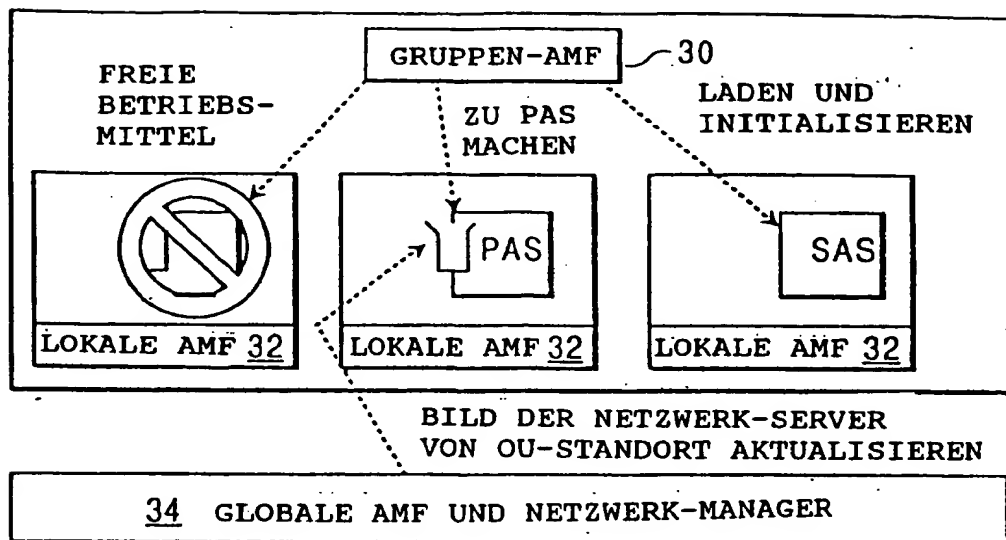


FIG. 3E

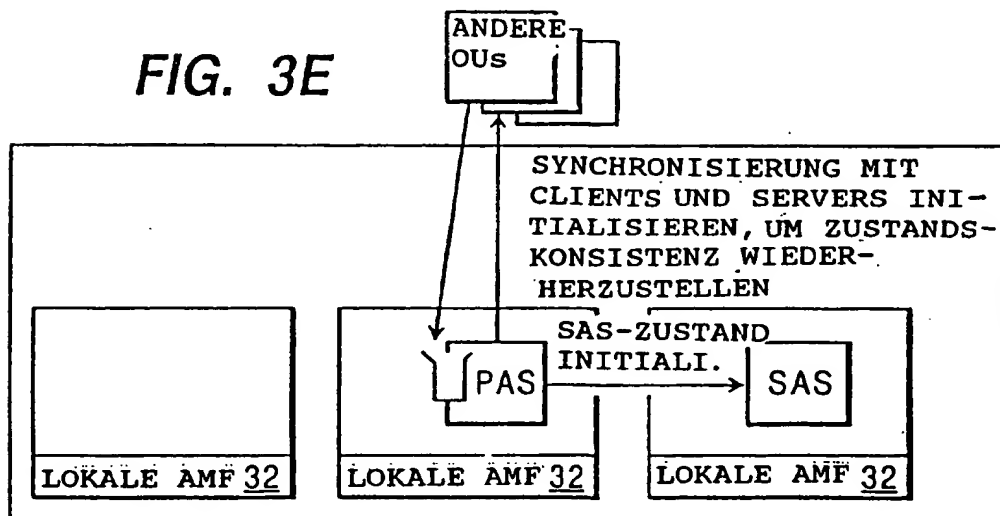
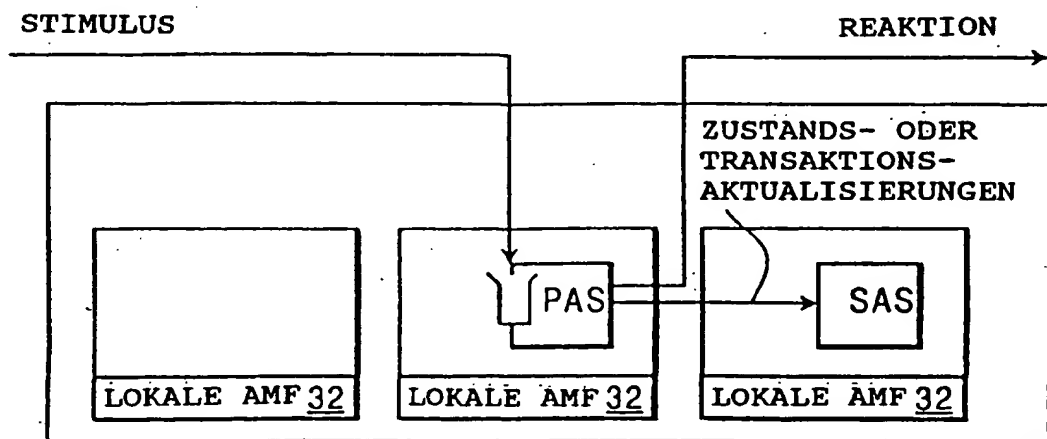


FIG. 3F



THIS PAGE BLANK (USPTO)